LAMP-TR-114
CAR-TR-998
CS-TR-4605
UMIACS-TR-2004-47

MDA904-02-C-0406
July 2004

Automatic Performance Evaluation for Video Summarization

Mei Huang, Ayesh B. Mahajan and Daniel F. DeMenthon

Laboratory for Language and Media Processing (LAMP)
University of Maryland, College Park, MD 20742-3275
may@umiacs.umd.edu
ayesh@umiacs.umd.edu
daniel@cfar.umd.edu

Abstract

This paper describes a system for automated performance evaluation of video summarization algorithms. We call it SUPERSIEV (System for Unsupervised Performance Evaluation of Ranked Summarization in Extended Videos). It is primarily designed for evaluating video summarization algorithms that perform frame ranking. The task of summarization is viewed as a kind of database retrieval, and we adopt some of the concepts developed for performance evaluation of retrieval in database systems. First, ground truth summaries are gathered in a user study from many assessors and for several video sequences. For each video sequence, these summaries are combined to generate a single reference file that represents the majority of assessors' opinions. Then the system determines the best target reference frame for each frame of the whole video sequence and computes matching scores to form a lookup table that rates each frame. Given a summary from a candidate summarization algorithm, the system can then evaluate this summary from different aspects by computing recall, cumulated average precision, redundancy rate and average closeness. With this evaluation system, we can not only grade the quality of a video summary, but also (1) compare different automatic summarization algorithms and (2) make stepwise improvements on algorithms, without the need for new user feedback.

Keywords:

Automatic performance evaluation, video summarization, frame ranking, average precision, TREC, ground truth

Support of this research by Hitachi Systems Development Laboratory under Contract 0210108898 and by the Department of Defense under Contract MDA904-02-C-0406 is gratefully acknowledged.

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
JUL 2004		**00-00-2004 to 00-00-2004**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
Automatic Performance Evaluation for Video Summarization	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)	8. PERFORMING ORGANIZATION REPORT NUMBER
University of Maryland,Institute for Advanced Computer Studies,Language and Media Processing Laboratory,College Park,MD,20742	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES
The original document contains color images.

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			
unclassified	**unclassified**	**unclassified**		**32**	

1 Introduction

Video summarization is a family of techniques aimed at

1. *Speed watching*: helping the user understand videos at a faster pace than the normal pace. This is a form of speed reading for videos.

2. *Browsing*: allowing the user to navigate the video at a higher altitude and ignore details, and dive down when a region of interest is detected.

3. *Query guided search*: allowing the user to see subsets of videos in response to queries such as "Show me all the goals in this recorded soccer game".

For each of these modalities, the summaries can chop the videos either at the level of shots or at the level of frames: in the first case, summaries take the form of video skims, in which some shots are dropped completely, while others are played with audio but may be truncated [17, 13, 7]; in the second case, they use individual key frames. The key frames can be presented to the user either in temporal form as a smart fast-forward function [1, 2], in spatial form as a storyboard [14], or in spatial-temporal form as a storyboard of animated thumbnails.

Methods that let the user set up the level of summarization (also called abstraction level [8], compression level [11], or altitude [1, 2]) of the video stream seem desirable as they provide more control and more freedom compared with those that decide for the user what the best level of summarization should be. A better solution seems to be a combination of these two views, where the system is initialized to an optimal default level when it is started; the user can explore other altitudes by dragging the slider of a scrollbar, and return to the default level by clicking a button [1]. This slider control can be implemented by ranking individual frames or shots. The evaluation system described in this paper was developed specifically for such *ranked summarization algorithms* that rank each individual frame [1, 2]. For more complete surveys on video summarization techniques, refer to [6, 8].

2 Related Work in Performance Evaluation

2.1 Video Summary Evaluation

Although video summarization has been the subject of very intensive research, to the best of our knowledge there is no standard method to evaluate its performance, largely because of the subjectivity of the evaluation process. Currently, there are three main types of evaluation methods described in the literature:

1. In the most elementary type, storyboards produced by the candidate algorithm and by a competing algorithm (such as an elementary summarization by equal-time-interval sampling) are shown in parallel figures, and the reader is asked to confirm that candidate algorithm has produced a better storyboard [2, 15]. It is evident that if there are more than two algorithms to be compared, it becomes harder for readers to rank them without quantifying their intuitive estimations against specified criterions.

2. In a more quantitative approach, a pool of assessors (often reduced to the author of the evaluated algorithm alone) review a given video, and designate specific frames (for key frame summarization) or shots (for video skim generation) as being important for the understanding of the video. Then the output key frames or shots of the algorithm are manually matched by researchers to those of the ground truth. The matches are counted, and some measures, such as precision and/or recall, are computed to grade the algorithms [6, 4, 12]. This grade can be used to compare the candidate algorithm to a previously published one [4].

3. Another quantitative approach uses psychological metrics and evaluates the cognitive value of summaries. Questionnaires are used to measure the degree to which users understand the original video when they only see the summaries. Subjects are asked to judge the coherence of summaries on a scale of 1 to n, and give their confidence in answering questions about the content of the video (who, what, where, when) on preset scales [13, 7, 19], or give their satisfaction when watching summaries compared with watching the whole video [19]. Summarization algorithms are then ranked on the grades received on these tests. This type of evaluation is complex if it is to be done right. For example, assessors may have some prior knowledge about the video content, so special questionnaires should be designed to address this, e.g., subjects are asked to answer questions *before* and after they see the summary [7]. Furthermore, it is hard to decide the appropriate number of grades or scales; in [19], three grades may be too coarse to reflect individual difference, while too many grades need very detailed guidelines and definitions, thus increasing the difficulty for assessors.

The above approaches are similar in three main aspects, which we see as their main drawbacks. First and foremost, they are designed just to tell whether the algorithm is good or not or is better or worse than another, and therefore they are not aimed at giving us hints on how to improve the algorithm, which we think is precious to researchers and should be included in the evaluation. Second, the time investment in user study for evaluating one algorithm is not reusable for evaluating another algorithm; all the effort has to be repeated each time an algorithm has been changed or a new algorithm has been developed. Finally, these approaches are manual, which is very time consuming and error-prone.

On the other hand, there are several well developed automatic evaluation systems in the fields of language processing, specifically in the areas of retrieval of relevant text documents [16], machine translation [10], and automatic text summarization [11]. Though they are not targeted at video summarization algorithms, their principles and techniques can be adopted and adapted to our purpose.

2.2 Evaluation in Text Related Fields

The principle of assessment and evaluation set up for the Text Retrieval Conference (TREC) is illustrated in Fig. 1.

In this approach, assessors each produce a reference file showing which elements of the data set are relevant or non-relevant. The algorithm to be evaluated is run against the same data set, but goes further than assessors in the sense that it ranks each element of the dataset from most relevant to the least. Then the output of the algorithm is compared to each reference file by incrementally going through the ranked list; a precision is computed each time a newly included

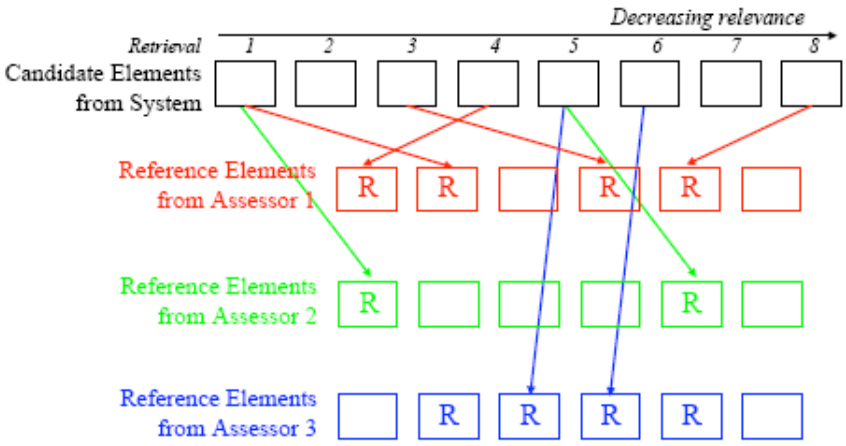


Figure 1: Principle of performance evaluation used for the Text Retrieval Conference (TREC). The algorithm under evaluation ranks elements of a data set in terms of their relevance to a query (top row). A number of assessors have individually marked the elements as relevant or non-relevant for that query to produce reference files (next three rows). The candidate ranking from the algorithm is compared with each reference file to produce an average precision, and these precisions are averaged over a range of data sets, queries and assessors to produce a mean average precision used as a global grade for the algorithm.

element is matched to a relevant item in the reference file. When a specified number N of elements from the ranked list has been considered, all the precisions are averaged to give the final mark. This protocol yields a grade for each comparison to a reference file. After repeating the process for many data sets, the average of the resulting grades, called the mean average precision (MAP) is used as a global grade (between 0 and 1) for the algorithm under study.

In TREC, the elements are documents, and the algorithms under study are designed for query-based retrieval of documents. An instance would be "Find all the documents about the independence of Quebec", and assessors were asked to mark the documents as relevant or non-relevant under this criterion. In the evaluation of the evaluation system itself, it was shown that assessors sometimes have non-overlapping opinions about which documents are relevant for specific queries, but that MAP averages out these differences and is a useful indicator of quality for algorithm improvement [16]. Clearly, the principles of the TREC evaluation can be applied to any system that ranks elements of a data set according to some criterion, provided that human assessors can be used to produce reference files under the same criterion.

The principle of BLEU (BiLingual Evaluation Understudy), a system developed at IBM for evaluation of machine translation [10], is similar to that of TREC, but the grading process is different: a single precision is calculated, using the rule that the relevant marking of an element by an algorithm is correct so long as there was one assessor who judged it as relevant; conversely it would be marked incorrectly if none of the assessors regarded it as relevant. In the case of machine translation, the data set is the whole vocabulary of the target language, and a specific translation can be interpreted as a file in which the words required to express the meaning of the translation are marked as relevant in the data set, while the other words are not relevant. The order of the translated word is not used in the evaluation; i.e., a scrambled translation gets

the same grade as one in a grammatically correct order. Also, there is no attempt to match words with their synonyms; only exact matches are used. Intuitively, the need for synonyms subsides as more translations are pooled as references and more variations for translating the same sentence are included. In spite of its simplicity, BLEU was shown to be useful for improving machine translation algorithms, even when few reference translations are used [16]. There were attempts to extend this framework to the domain of automatic text summarization, with mixed preliminary results [11]. Note that in BLEU, the algorithm is asked to perform the same task as the human assessors (mark elements of the data set as relevant). Therefore with BLEU one can use a human assessment and use it as if it came from a machine. This was useful to show that BLEU is consistently giving better grades to humans than to machines, a comforting sanity check. Such tests are not possible with TREC-type evaluation systems, since the element ranking by algorithms is required for MAP estimation, while assessors are not asked to rank the data set elements (it would be too much work). Even with this advantage for BLEU, the average precision of the TREC approach is more appropriate for evaluating ranked summarization algorithms for videos than the basic precision used by BLEU, as will become clear in Section 4.

2.3 From text to Video

By viewing video frames as the elements of the data set composed of the whole video, and viewing video summarization as a kind of data retrieval task, the principles used in the above text related areas can be borrowed for video summarization evaluation. In this paper, we propose an automatic evaluation approach, which makes the user study easier to conduct and the results reusable and can give a relatively clear view about the merits and shortcomings of the summarization algorithms, so that evaluation can be included in the software development process of algorithms. The system designed for this purpose is called SUPERSIEV (System for Unsupervised Performance Evaluation of Ranked Summarization in Extended Videos).

The remainder of this paper is arranged as follows: Section 3 gives a short introduction of curve simplification based frame ranking algorithm; Section 4 describes system design and performance evaluation measures; Section 5 describes the evaluation process and Section 6 introduces ground truth collection and processing through user study. We show our experiment results in Section 7. Conclusions and future work are in Section 8.

3 Frame Ranking Using Curve Simplification

To illustrate the operation of SUPERSIEV, we describe its application to a video summarization algorithm using curve simplification [1]. In this algorithm, each frame is mapped to a point in a high dimensional space, so that the video stream is mapped to a trajectory. The mapping is selected so that the trajectory is linear when the video is static, and displays a high curvature when changes occur. Then a filtering operation using an iterative fine-to-coarse polygon simplification mechanism is applied (See Fig. 2), which ranks the frames according to their contribution (relevance) to the shape of the video trajectory. The relevance measure $\boldsymbol{K}$ of vertex v is defined as: $\boldsymbol{K}(v) = \mathrm{d}(u, v) + \mathrm{d}(v, w) - \mathrm{d}(u, w)$, where d() is the distance function and u, w are two neighbor vertices of$v$. The more the length of the polyline is decreased when a vertex v is removed, the higher its relevance is.

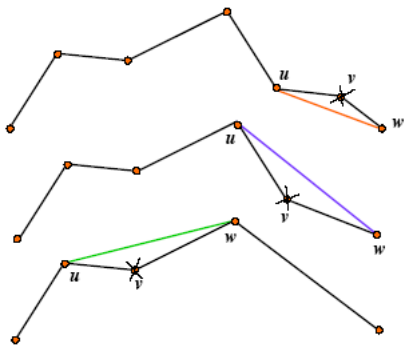


Figure 2: Three steps of fine-to-coarse polygon simplification, where at each step the vertex removal that produces the minimal reduction in the total length of the polygon is selected. A removed vertex is indicated by a crossed-out point and a line segment bypassing the point. Note that the points removed first are those most aligned to their neighbors. The mapping insures that such points correspond to predictable frames. The order of removal provides a ranked list of video frames from least relevant to most relevant. The process is stopped when only the end points remain.

We apply the automatic evaluation system to compare the performance of the distance function using the following feature vectors, which use the same construction scheme with different color components combinations.

- Basic YUV (bYUV for short): We define four histogram buckets of equal size for each of the three color attributes in the YUV color space of MPEG encoding. Each bucket contributes three feature components: the pixel count and the 2D spatial coordinates of the centroid of the pixels in the bucket.
- Basic Y (bY): same as Basic YUV, with removal of U and V information.
- Smooth YUV (sYUV): There could be occurrences when a large population of pixels falls in one bin, but very close to the border with a neighbor bin. Illumination flickering could make this population jump to the neighbor bin, producing a spike in the trajectory and an erroneous high relevance. To make the histogram less sensitive to noise, we smooth a 256-bin histogram by Gaussian convolution to make the large peaks spread out over several bins and then combine bins of the smoothed histogram to form the final histogram as in Basic YUV.
- Smooth Y (sY): same as Smooth YUV, with removal of U and V information.
- 3DYUV: This construction considers each pixel to be a 3D point with Y, U, V components. Each bin has the same three components as in other constructions.
- logRGB : Here we apply Slaney's technique for constructing a 3D histogram [12]. The total bin number is 512, and each bin index is expressed using 9 bits.

$$\boldsymbol{I}_{bin} = \mathbf{64\ floor(\log_2(R))\ +\ 8\ floor(\log_2(G))\ +\ floor(\log_2(B))}$$

The advantage of this logarithmic scaling is that it is found to better spread bin counts among the 512 bins [12]. For each bin, we compute the same three components as in the other constructions.

4 System Design

Before we describe the evaluation system's architecture and performance measures for video summarization algorithms, a key issue we must face and solve is how to define a good summary in general. With algorithm improvement as our final goal, we need to compare the outputs of candidate algorithms with some benchmark summaries, which should be optimal ones. We think summaries from human assessors can be used as the benchmarks, because, for one thing, video summaries are targeted at human users and people know what a good summary is and how to make a good summary; second, we do not believe and do not expect that computers can outperform general human assessors in understanding video content.

Therefore, we define a good video summarization algorithm as one producing summaries that match human summaries well. For improvement purposes, it is preferable to break this general definition into detailed aspects, i.e. to define a "good match" from different view angles, so that it will be clear to researchers which aspects of the algorithm contribute to its overall performance after evaluation. Specifically, a good match implies four aspects:

1. As many frames in the reference summary are matched by the candidate summary as possible;
2. As few frames in the candidate summary do not match reference frames as possible;
3. As few frames in the candidate summary match the same reference frames as possible;
4. Each matched frame in the candidate is as similar to its corresponding target frame in the reference summary as possible.

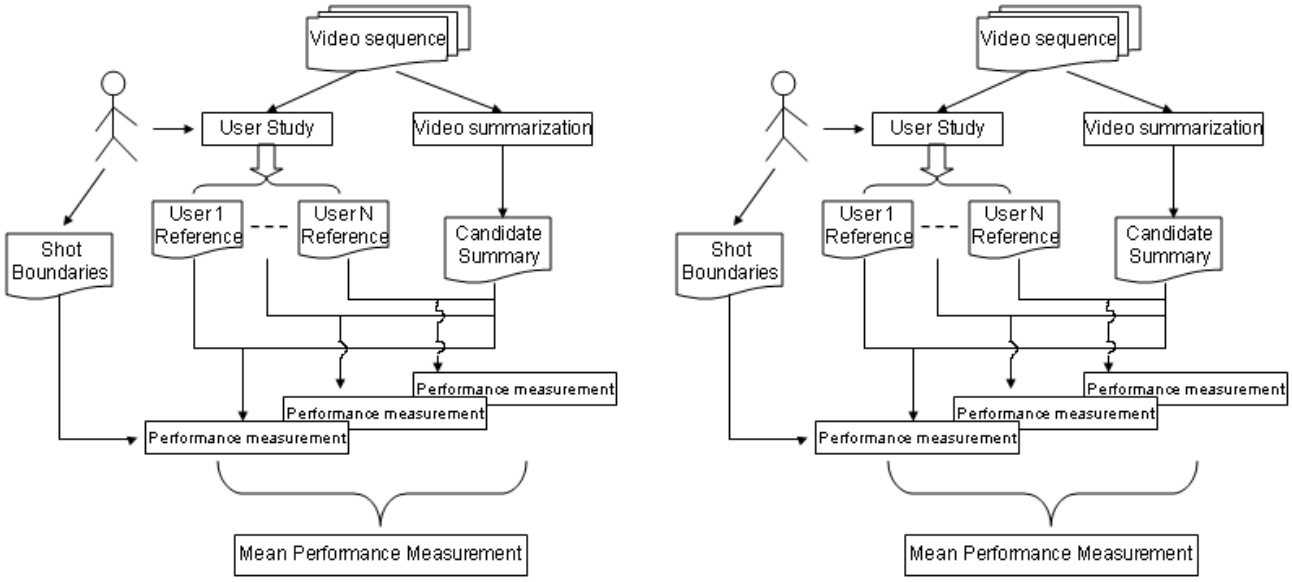
Generally, the former three aspects are necessary, and (4) is a reinforcement of (1). But these are not always sufficient requirements. For specially designed video summarization algorithms, additional aspects should be considered, e.g. for frame ranking algorithms such as [1], one would also like to see whether the top ranked part of the output frame list is enjoyable and informative enough (See Section 4.2.2).

Once we have reached an agreement on what makes a good summary, we can build our system architecture and design measures to evaluate the given candidate summary from different aspects.

4.1 System Architecture

We have collected a varied corpus of video sequences for our experiments. First, we use the user study tool (see Fig. 10) to generate files containing the frame indices of the shot boundaries. Then assessors (User 1 to User N in the figures) produce reference summaries for each sequence of the corpus with the help of the user study tool. Then every time a new algorithm is tested, it is run with the set of video sequences and its performance is the set of performances on each video.

Intuitively, there are two possible schemes to evaluate a candidate summary against the reference summaries from assessors. One is to compare the candidate summary with each of the reference file and compute the performance measures for each comparison; As Fig. 3(a) shows, the final performance is the average of the scores over all reference summaries. The other, as shown in Fig. 3(b), synthesizes reference summaries first to get a final reference file, and then evaluates the candidate summary against this synthesized reference.



(a) (b)

Figure 3: Two possible schemes (a) Compare the candidate with each of the N references and compute the mean performance. (b) Synthesis the N references into one and compute the performance against this synthesized one.

When the number of assessors is small, the advantage of synthesizing is not so significant. However if there are many assessors, through synthesis a large amount of time can be saved by avoiding repeating the same frame pair matching procedure. In addition, we argue that synthesis can lead to a more reasonable and efficient performance evaluation (See Section 6.1) and we adopt this scheme in SUPERSIEV.

4.2 Performance Measure Design

We adopted concepts developed in database systems for evaluating data retrieval, specifically techniques developed for the TREC conference. We regard the summarization process as a type of database retrieval, with the data consisting of all frames in the target video and the reference summaries generated by assessors forming the ground truth. Reference files specify which frames are important for a summary (***Class A*** in Fig. 4) and which are not (***Class B***). From this view angle, the aspects (1)(2)(3) mentioned above as the requirements for a good summary can be

interpreted as ***high recall***, ***high precision***, and ***low redundancy*** respectively.

It is evident that the comparison of two frame lists is based on the comparison of single frame pairs, thus we need define some distance measure between frames first. Clearly, within shots, two frames with exactly the same index can be safely matched, but in addition, two frames with different indices could also be matched if their visual content is similar. For long static shots, two frames could be several seconds apart in the video and still be good matches. Therefore, we need to define a measure of dissimilarity between frames characterizing the distance between visual content of the frames. In our experiment, we represent each frame as a thumbnail, by applying a 2D Gaussian smoother to the original image, and line-scan all pixels to form a high dimensional vector. Distance between frames is computed as the Euclidean distance between these feature vectors, although the proposed performance evaluation scheme could accommodate other distances as well.

In the following subsections, we describe how we adapted conventional performance measures to our purpose of evaluating curve simplification video summarization algorithms.

4.2.1 Recall

In our settings, the goal of recall is to measure how many frames in the ground truth are matched by the candidate summary. A reference frame is matched if there exists at least one frame in the candidate summary that is similar to the reference frame according to some preset criterion.

Recall is computed as:

$$\text{Recall} = \boldsymbol{N}_{ref\_m} \;/\; \boldsymbol{N}_{ref}$$

where $\mathbf{N}_{ref}$ is the total number of frames in a reference summary, and $\mathbf{N}_{ref\_m}$ is the number of reference frames being matched.

4.2.2 Precision, Average Precision and Cumulated Average Precision

Intuitively, precision is a means of measuring how many frames in the candidate summary match frames in the reference summary, and it can be defined as

$$\text{Precision} = \boldsymbol{N}_{can\_m} \;/\; \boldsymbol{N}_{can}$$

where $\mathbf{N}_{can}$ is the total number of frames in a candidate summary; and $\mathbf{N}_{can\_m}$ is the number of candidate frames being matched. Since one frame in the reference summary might be matched by several frames in the candidate summary, $\mathbf{N}_{can\_m}$ can be computed as equal to $\mathbf{N}_{ref\_m}$ or include redundant matches in the counting. In our system, we choose the former to only count the significant matches.

However, this measure is not fit for video summarization algorithms whose output is the ordered frame list of the whole video, such as curve simplification frame ranking algorithms. In such cases, as we go through the list, recall surely will reach 100% at last and precision is a constant $\mathbf{N}_{ref}$/ $\mathbf{N}_{v}$for all candidate summaries from the same category of algorithms, where $\mathbf{N}_{v}$ is the number of frames of the video. To address this problem, one can compare the precisions at a specific point, such as when recall is 100% and now the precision is computed as $\mathbf{N}_{ref}$/ $\mathbf{N}_{100}$, where $\mathbf{N}_{100}$ is the number of frames of the candidate summary that the evaluation system goes through for recall

to reach 100%. Since $\mathbf{N}_{ref}$is the same for all candidates, precision can be interpreted as the time cost of reaching a certain degree of recall.

Is this enough for comparing such algorithms? Let's look at an example. As Fig. 4 illustrates, the reference summary is composed of five frames (which are matched by ***Class A*** frames) and the Case 1 candidate summary and Case 2 candidate summary both use eight frames to reach 100% recall, i.e. precision is the same for both cases (***P = 5/8 = 0.625)***. Does this mean the two have equal performance? For frame ranking video summarization algorithms, one would also like to evaluate the ranking performance without obtaining ranked reference summaries. We can measure the ranking quality in a more general way. As Fig. 4 shows, it is easy to tell that the candidate summary in Case 1 is better than that in Case 2, because high relevance frames in the first candidate correspond to reference frames.

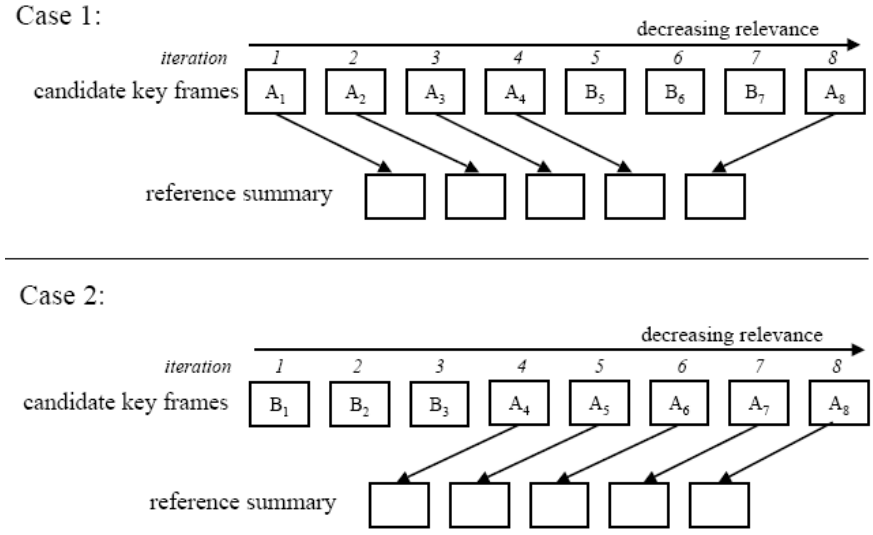


Figure 4: Evaluation of video summarizers that rank frames by relevance. In Case 1, frames of highest relevance correspond to reference frames. In Case 2, one has to go further down the ranked list to retrieve all the reference frames.

Based on this observation, we see that the precision at the point when recall reaches a certain degree in the first time is not enough to reflect the quality of the whole ranked list, and turn to Average Precision (AP for short), which is the averaged value of the precisions at all iteratively increased numbers of ***Class A*** frames.

$\mathrm{AP(k)} = \sum_{k=1}^{Nref\_m} p(k)I(k)/\boldsymbol{N}_{ref\_m}$

where p(k) is the precision at the $\boldsymbol{k}^{th}$frame in the candidate summary, I(k) is an indicator which is 1 only when frame k causes the increasing of recall, i.e., is a ***Class A*** frame and brings a new frame match.

The previous example shows that AP captures the requirement for the candidate summary of Case 1 to receive a higher mark than that of Case 2. For Case 1, a new ***Class A*** frame is found in iteration 1, 2, 3, 4 and 8, and its average precision is

$$\boldsymbol{AP}_1 = (\mathbf{P}_1 + \mathbf{P}_2 + \mathbf{P}_3 + \mathbf{P}_4 + \mathbf{P}_5)/5 = (1/1 + 2/2 + 3/3 + 4/4 + 5/8)/5 = \mathbf{0.925}$$

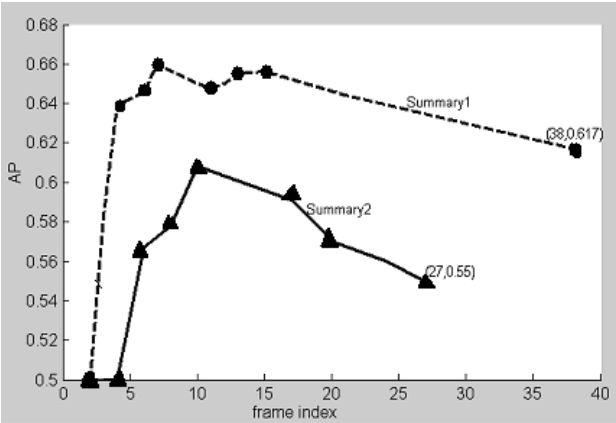


Figure 5: AP-frame index plot of two ranked frame list for video FlyingCar.mpeg. *Summary 1* uses 38 frames to reach 100% recall, and has a much higher AP than *Summary 2*, which uses 27 frames to reach 100% recall.

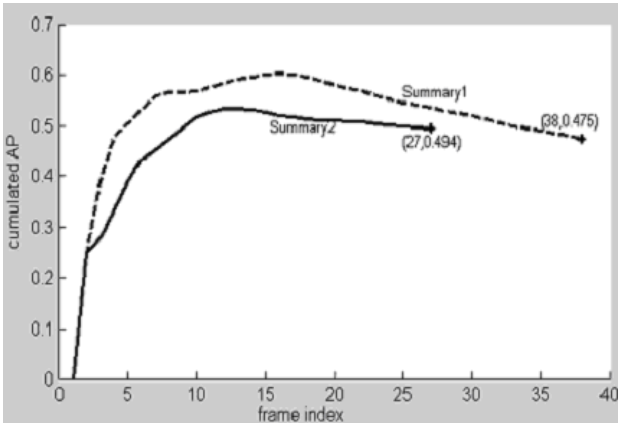


Figure 6: CAP - frame index plot of the two candidate summaries in Fig. 5

On the other hand, for Case 2

$$\boldsymbol{AP_2 = (P_4 + P_5 + P_6 + P_7 + P_8)/5 = (1/4+2/5+3/6+4/7+5/8)/5 = 0.469}$$

A high AP indicates that the high relevance frames in a candidate summary match reference frames. Note that AP must be used with $\mathbf{N}_{100}$to give a global view, i.e. AP alone can lead to some biased evaluations. Fig. 5 shows an example where AP of *Summary 1* is higher than that of *Summary 2*, yet it takes an evidently longer time for *Summary 1* (38 frames) to reach 100% recall than the latter (27 frames). And for applications where shorter summary length is appreciated more than ranking, AP alone cannot be a good measure.

We can also combine AP and $\mathbf{N}_{100}$ to form a new measure, which, in some cases, is more convenient. We define cumulated AP (CAP) of frame $\boldsymbol{i}$ in candidate summary as:

$\mathrm{CAP(i)} = \sum_{k=1}^{i} p(k)/i$

where $\boldsymbol{p(k)}$ is the precision at the $\boldsymbol{k}^{th}$frame in the candidate summary. Fig. 6 shows the CAP for the same two summaries.

4.2.3 Average Redundancy Rate

Redundancy in a video summary means that some frames are repetitive; i.e., the information provided by some frames can be derived from other frames. For our evaluation, redundancy for one reference frame can be measured by counting how many frames in the candidates can match this frame. For the whole candidate summary, we define Average Redundancy Rate as:

$\mathrm{AR} = \boldsymbol{N}_{can\_v} \;/\; \boldsymbol{N}_{ref\_m}$

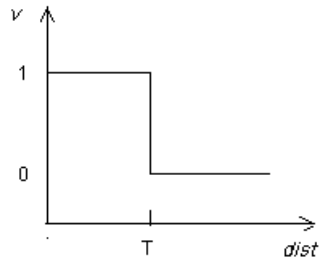


Figure 7: 0-1 match for a single frame pair

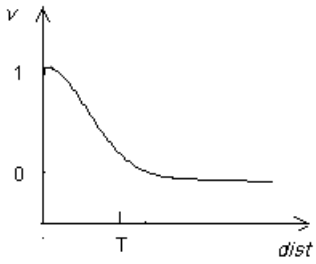


Figure 8: Probability match for a single frame pair

where $\boldsymbol{N}_{can\_v}$ is the number of frames in the candidate summary that can match at least one frame in the reference summary. For frame ranking based summarization algorithms, $\boldsymbol{N}_{can_v}$is no more than N_{100}.

4.2.4 Average Closeness

For computing recall, one should firstly decide whether a frame pair is a match. This assertion involves a thresholding procedure, classifying pairs as match or non-match according to some preset distance threshold. A usual way to do this is as Fig. 7 shows, a typical 0-1 classification. However, in some cases, people may feel that it is insufficient to know whether the candidate summary matches the reference; they may also want to know how similar the candidate summary is to the reference in visual content. This information is useful for comparing two summaries with similar or equal scores in all other performance measures. One way to do this is to use a continuous mapping function, which can reflect the individual differences when distance is less than a threshold and assign zero when distance is larger than the threshold, as Fig. 8 shows.

After we get a score for each single matching pair, we can compute the matching quality of the whole summary as the averaged score over all valid individual scores that cause an increase of recall. We call this measure Average Closeness:

$$\mathbf{AC} = \sum_i^{Nref\_m} \boldsymbol{V}(\boldsymbol{d}_i) \ / \ \boldsymbol{N}_{ref\_m}$$

where $\boldsymbol{d}_i$ is the distance between the $\boldsymbol{i}^{th}$ matched reference frame and its corresponding candidate frame; and $\boldsymbol{V(d}_i\boldsymbol{)}$ is a map function with range [0,1].

So far, we have discussed the four aspects mentioned in the beginning of Section 4 and designed performance measures for them. In the following sections, we describe the evaluation process and the reference gathering and processing procedures.

5 Evaluation Process

After reference synthesis, we now can evaluate a candidate summary by single frame matching test and sequence alignment.

5.1 Continuous Scoring Function

As mentioned in introducing the performance measure Average Closeness, the typical 0-1 classification enforces a discontinuity and will lead the left and right neighbor frames in the distance dimension of the frame corresponding to threshold T to belong to different classes (see Fig. 7), even though these two frames are very much alike and are probably neighboring frames along the time axis. So the enforced discontinuity in assertion contradicts human judgment. Fig. 8, on the other hand, effectively "softens" the classification procedure by applying a continuous matching score function of image distance, thus we can compare two frame pairs in more details.

For choosing such a continuous function, its desirable properties should be first determined. Obviously, it should be a function of image distance, and decreases as distance increases. But a simple reciprocal function is not suitable because it is infinite for zero distance. The function should have the following properties:

1. continuous;
2. mapping the input distance interval [0, ∞] to [1,0].

These properties let one think of the exponential family. In SUPERSIEV, we define the mapping function as $e^{-dist(i,a)^2/\lambda^2}$, where ***dist(i,a)*** is a distance metric, $\boldsymbol{i}$ and $\boldsymbol{a}$ are indices of frames whose matching degree is to be estimated. λ is an adjustable parameter, reflecting the subjective strictness exerted in the scoring procedure. One way to choose λ is to define a reference matching score for two randomly selected images and then to do a parameter estimation. In our experiments, we set $\lambda = 10$. The value itself does not contain much intrinsic information; it is meaningful and useful only in comparisons.

5.2 Sequence Alignment

The scoring function defines the score of a single match; to compute the average closeness of a whole candidate summary, which is a list of frames, an important step is to determine the matched sequence in a reasonable way. There are two possible ways: one way is using dynamic programming alignment to get the sum of the matching scores of frames inside each shot to reach a maximum; the other is simply a greedy approach consisting of selecting the optimal single frame, if it exists, for each reference frame. We describe the two methods in the following subsections respectively and argue that the latter is better in that its resultant alignment is more consistent with people's opinion.

5.2.1 Dynamic Programming Alignment

Sequence alignment problems can be solved optimally by dynamic programming, if the cost function is defined as the sum of distance of each aligned frame pair and the goal is to find an alignment to minimize the cost. A method similar to the Needleman-Wunsch algorithm used for aligning elements of amino-acid sequences in biology [9] can be applied. This involves an iterative matrix computation. The matrix elements correspond to frame pairings between the $\boldsymbol{m}$ frames of $\boldsymbol{S}_1$ (rows) and $\boldsymbol{n}$ frames of $\boldsymbol{S}_2$ (columns) and each of these matrix elements contains the cost of the cheapest pathway up to that pairing.

All possible alignments are represented by pathways in this matrix. Because temporal inversions are not allowed, only pathways going downward from left to right are acceptable. The two main steps in the algorithm are:

1. For each element of the current column, consider all the pathways that can reach this element, and select the least costly pathway. This operation involves only scanning the values of the previous rows in the previous column. Keep its cost in the current matrix element. Keep the index for the least costly pathway link between the previous column and the current one in a separate pathway matrix.

2. When the last column has been completed, select the pathway that reached that column with the lowest cost, and use the pathway matrix to recover the whole pathway working backward from the last column to the first column.

However, is this cost function and its minimum really consistent with our expectation? In the following example, ***ABC*** are three consecutive frames in the reference, and ***abc*** are three consecutive frames in a candidate. Suppose that ***a*** is within distance threshold to both ***A*** and ***B***, and ***b*** and ***c*** can only be matched to ***B*** and ***C*** respectively. Then by dynamic programming alignment, ***abc*** are matched to ***ABC*** respectively to reach a minimum summed distance for ***ABC*** and at the same time to reach a maximum recall. But, if ***a*** is much more similar to ***B*** than to ***A,*** the intuitive judgment would lead to such an assertion that ***a*** and ***b*** are both matched to ***B*** with redundancy and ***A*** is unmatched. So in this example, we see that the evaluation is not consistent with human judgment.

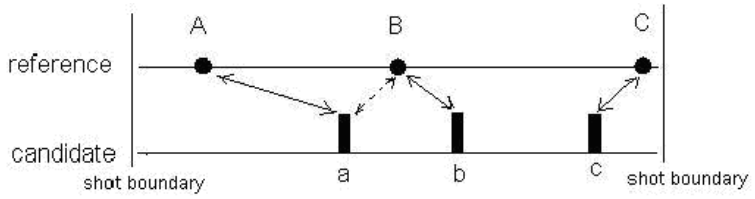


Figure 9: Biased evaluation introduced by dynamic programming alignment

5.2.2 A Simpler Way

Learning from the above example, we propose a simpler way to determine the final matching sequence. That is, looking through each frame in the candidate summary one by one, we first determine the potential matching targets for them. By enforcing temporal order, each frame can have at most two target reference frames, which are the nearest left and right reference frames in the same shot. Then the final matching target is determined by choosing the most similar one. For example, in Fig. 9, frame ***a*** has two potential matching targets, ***A*** and ***B***, and its final matching target is ***B*** because the distance between them is shorter. After obtaining these matched frame pairs, we go through each reference frame: if there is more than one candidate frame matched to it, we select the one that has the highest matching score as the optimal match and use it to compute the performance measures. To avoid time-crossing matching, special examination should be conducted. For the two frames involved, we remove the pair matching that has the smaller matching score and keep the one that has the larger matching score.

5.3 Offline Lookup Table

When the number of candidate summaries needed to be compared is large, there would be a lot of repetitive frame pair matching tests, which degrades the system efficiency. Here we propose to use an offline lookup table to accelerate the online evaluation procedure. Given a reference summary, the matching target and matching score of each frame in the video sequence can be determined shot by shot. Specifically, the lookup table of one shot can be computed in the following steps:

1. locate the potential targets of each frame $\boldsymbol{i}$, which are its neighbors along the temporal dimension in the reference of the same shot;
2. for each potential target of frame$\boldsymbol{i}$, compute its matching score using the continuous scoring function mentioned above;
3. if frame $\boldsymbol{i}$ has two potential targets, the one that gets the larger matching score is chosen as the final target of frame $\boldsymbol{i}$, and this target's frame index and the matching score are recorded in the lookup table.

After the lookup table is obtained, the online matching procedure for a single frame in the candidate summary only consists of retrieving the corresponding table entries by frame index.

6 Ground Truth Collection and Processing

6.1 User Study Interface

We have developed software for the collection of reference summaries from assessors. Its user interface is shown in Fig. 10. This tool provides functions for convenient video browsing and individual frame capturing for storyboard production and shot boundary extraction. After loading a video sequence, the user can display the video continuously by clicking the playback button, or make fast jumps by dragging the slide bar, or display one frame at a time in forward or backward direction by using the arrow keys. While browsing, the user can capture frames and transfer them to the storyboard by clicking the **Capture** button. The frame thumbnails in the storyboard are always arranged in the temporal order no matter how they are selected. The storyboard constructed by the user is located at the right hand side of the video player (Fig. 10). Users can also make corrections to the storyboard by selecting a thumbnail and clicking the **Delete** button. The storyboard can be played back in the video player at a speed specified in the **Preferences** dialog box. This interface allows users to create the storyboard in a single or multiple sittings, and the storyboards can be saved and reloaded in a later time. Shot boundaries can also be located manually in the same way with this interface. Our evaluation system then uses assessor summaries in conjunction with reference shot boundary files to evaluate the performance of candidate algorithms.

6.2 Guidelines for Assessors

Assessors should be given some freedom on their summarization behaviorbut for our purpose of using assessors' summaries as ground truth, we need our assessors to do a good job. So it is necessary to first make it clear to our subjects what a good summary is.

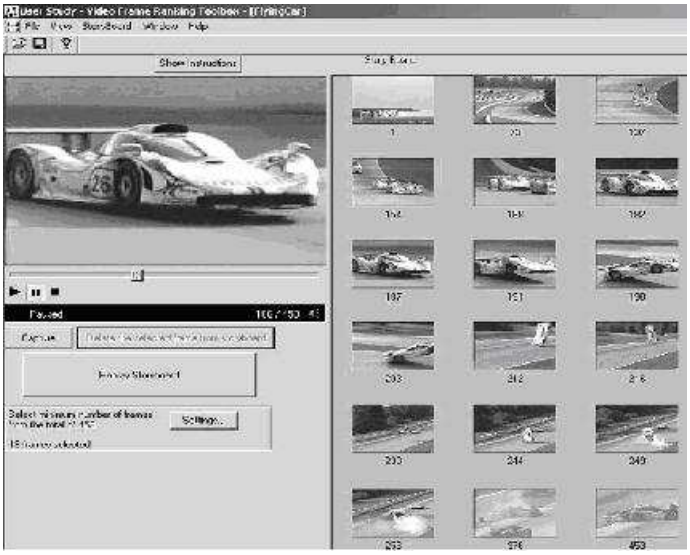


Figure 10: User interface for collection of reference video summaries

Generally speaking, a summary should be concise and keep the main points of the content of the original material. This is very straightforward, yet still too general to allow users to grasp the appropriate abstraction level. So we give detailed guidelines as follows. A summary must be:

1. as terse as possible so long as the completeness of main content be preserved
2. as neutral as possible, to let the viewers themselves discern the abstract meanings

For the first guideline to be well carried out, we suggest to our subjects that they summarize only after viewing the whole video so that they can get an overall impression about the content. We also suggest that after summarization they replay the summary in the user study interface (see Fig. 10) to refine it, for example by deleting non-important and repetitive consecutive frames.

The second guideline is in consideration of the neutrality of automatic summarization algorithms. People from different background may have different viewpoints about the same video, so we ask subjects that they try to exclude their personal preferences as much as possible, suggesting that they put themselves in a position of making a summary for general audiences. This is for general summarization algorithms; if an algorithm is aimed at generating summaries for special audiences, this should be adapted correspondingly.

6.3 Reference Synthesis

Reference synthesis is aimed at cases when there is no expert to help produce a ***decisive*** reference summary. The idea is to extract a single mainstream summary among all subjects of the user study and use this as the benchmark, so that the candidate summary that matches the benchmark exactly is taken as optimal.

6.3.1 Motivation

As we mentioned before, there are two ways to evaluate a candidate summary against a pool of reference summaries from assessors, and we argue that synthesizing references can lead to a more reasonable evaluation besides its time efficiency. Here we point out the shortcomings of the scheme based on N comparisons and performance averaging, as shown in Fig. 3(a), to demonstrate the advantage of reference synthesis.

One problem is illustrated in Fig. 11. In this case, reference S_1 selects frame A, B, E for a shot, and reference S_2 selects A and E for this same shot. Suppose candidate summary Conly selects D for the shot and D is within the distance threshold from A and B, but is not similar to E. Then, when C is evaluated against S_1, D is matched to B due to the enforcement of time order or probably due to its shorter distance to B than to A; while when C is evaluated against S_2, D is matched to A. This seems not to affect the result of recall and AP or CAP, but will affect average closeness. And the inconsistency of D matching different reference frames for different references will render the performance averaging procedure unreasonable.

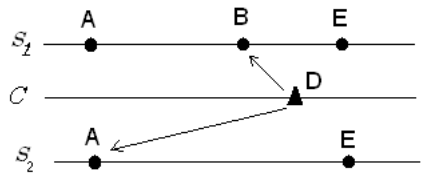


Figure 11: Inconsistence of matching target leads the averaged performance unreasonable

The second problem is with the rationality of the averaging procedure. By averaging performance results over all individual comparisons against each reference, the final performance favors a candidate that compromises to all references under the criterion of minimum summed distances. In Fig. 12, the line is some kind of projection axis (e.g. a performance measure); the filled black circles represent the reference summaries; the circled 2 represents their mean; and the circled 1 represents a local mean. When the circled 2 is viewed as the optimal, the candidate summary (represented as a triangle) that is close to it is better than the one that is near to the circled 1. However, it is impractical and not our aim to design an auto-summarization system that can cater to all users; our goal is to generate a video summary that matches the opinion of the majority of users. Thus for this purpose, it is clear the circled 1 is better than the circled 2 as the optimal.

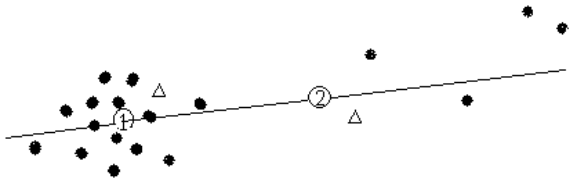


Figure 12: A good summary under mean performance criterion may deviate far from the majority of opinions

To deal with the above two problems, we adopt the second evaluation scheme (Fig. 3(b)), i.e.

use a synthesis of all the reference summaries that can reflect the mainstream opinion among all assessors, and then evaluate the candidate summary against this synthesized summary.

6.3.2 Reference Synthesis

Reference synthesis is based on two assumptions: (1) Each assessor has his/her constant semantic gap between consecutive summary frames; (2) Consistent common judgment among assessors exists (see Appendix A). The process involves three main steps: the first is to decide the number of clusters inside one shot based on a summary length histogram; the second step clusters reference frames in time dimension to get initial cluster boundaries and locates frame intervals that have local density maximum to obtain initial cluster centers. Then cluster boundaries are revised by comparing image distances from boundary feature vectors to the initial cluster center ones. The last step is to iteratively revise clusters after each time synthesizing in feature space each assessors reference summary frames, if there is more than one, inside each cluster into one. We formalized the detailed steps in Fig. 13. The final output synthesized reference is composed of the cluster centers arranged in temporal order.

The histogram {h(k)} used for deciding the expected summary length (Ec) for a given shot is illustrated in Fig. 14, whose bin index is the summary length and each bin counts the number of assessors whose summary length equals the bin index. And Ec is determined by the median value of the histogram. Note that the number Ec indicates that at least half of the assessors agree that the summary should be no longer than Ec. It is not that 50% assessors vote for this number; such high agreement would be very unlikely. Though Ec defined in this way may sometimes deviate far from the maximum single bin which is far left to the median, generally, the median value represents a relieving scheme. And the reason why we do not use the averaged summary length over all assessors is to reduce the effect of the minority outliers, a problem similar to Fig. 12.

The k-means clustering step is first conducted using the 1D time coordinate, i.e., along the time dimension. The following window searching process is to relocate initial cluster centers as where the sum of assessors reference frame within a local window with predefined size $\boldsymbol{w}$ (in our experiment, we set $\boldsymbol{w} = 5$) reaches a maximum. Then for each consecutive cluster pair, their neighboring boundaries are updated by reclassifying frames using the image distances to cluster centers. Note that the initial clustering is not conducted in feature vector space; for one thing, it is difficult to combine temporal order information in feature vectors; for the other, the output summary only needs the information of cluster centers. The iteratively revised centers are rather robust.

If there are enough assessors, a removal step of evidently abnormal references (minority cases that choose too few or too many key frames for a shot compared with the majority of others) can be applied to get rid of outliers at a coarse level, before the iterative cluster revision begins. This is helpful, because it is inevitable in a user study that some users would interpret the guidelines quite differently from most others due to personal reasons or a misunderstanding of our guidelines.

In our experiment, a total of 36 subjects participated in the user study. They were students with various majors, such as arts, electronic engineering, environment, computer science, etc.

Notation:

Na := number of assessors
T : = preset threshold, larger than 1
Ec := expected number of clusters
C_j := center of cluster j
R_j := right boundary of cluster j
L_j := left boundary of cluster j
$N(i, a_i)$:= number of reference frames of assessor a_i in shot i
$N(i, a_i, c_j)$:= number of reference frames of assessor a_i in cluster c_j of shot i
$I(a_i, t)$:= indicator, equals 1 when frame t is in the reference of assessor a_i, otherwise 0
$D(x,y)$:= distance between frame x and y in feature space

Main part:

For each shot i
  Make histogram $\{h(k)\}$ of summary lengths, as Fig. 12 shows;
  For k=1 to max (summary length for this shot)
    If ($\sum_{j=1}^{k} h(j) \geq Na/2$ and $\sum_{j=1}^{k-1} h(j) < Na/2$)
      Ec = k; Break;
    Endif
  Endfor
  Initial k-means clustering of all reference frames into Ec clusters using frame indices;
  For each cluster c_j
    Look at every w frames and Compute $S_t = \sum_{Na} \sum_{t}^{t+w} I(a_i, t)$;
    Find t= arg $\max(S_t)$ and locate the densest point in [t,t+w] as C_j ;
  Endfor
  For j = 2 to Ec
    If ($D(R_{j-1}, C_j) < D(R_{j-1}, C_{j-1})$ || $D(L_j, C_{j-1}) < D(L_j, C_j)$)
      Revise boundaries for c_{j-1} and c_j frame by frame;
    Endif
  Endfor
  For each assessor a_i
    If ($N(i, a_i) > T * Ec$)
      For each class c_j
        If ($N(i, a_i, c_j) > 1$)
          Average these $N(i, a_i, c_j)$ feature vectors into f';
          Find frame f_o that is nearest to f' in feature space and remove others;
        Endif
      Endfor
    Endif
  Endfor
  Redo Clustering
Endfor

Figure 13: Pseudocode for references synthesis

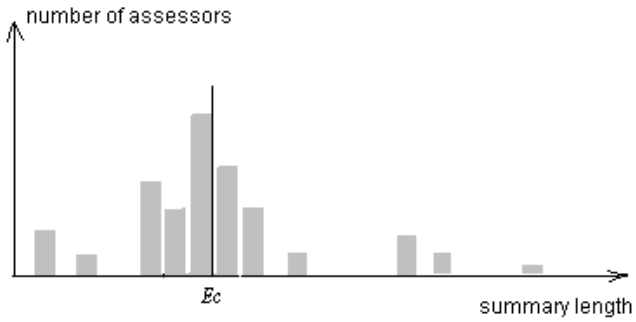


Figure 14: Histogram used to decide the summary length of a shot

7 Experiments and Discussion

7.1 Videos used in Experiments

We chose three videos from different genres for our experiments. They are:

- *FlyingCar.mpeg*, composed of 2 shots, is a sports news story full of quick action. The whole length is 451 frames, and the length of the synthesized reference summary is 12 frames.
- *Bebop.mpeg*, composed of 20 shots, is a Japanese animation video, with a large amount of nearly static frames. The whole length is 3114 frames, and the length of the synthesized reference summary is 24 frames.
- *WizzardDies.mpeg*, composed of 17 shots, is extracted from the movie *Lord of the Ring*. The whole length is 1024 frames, and the length of the synthesized reference summary is 19 frames.

7.2 Algorithms Evaluated

We have evaluated three summarization algorithms: ***Curve Simplification Frame Ranking Algorithm***, ***Time Bisection Algorithm*** (**Tbisec** for short) and ***Distance Bisection Algorithm*** (**Dbisec** for short). The output summaries of these three summarization algorithms are frame ranking lists.

Tbisec represents a video sequence in a hierarchical way by building a binary tree; each branching cutting the sequence of the parent into two subsequences of equal length, and the two children contain the lower and upper half of the sequence of the parent respectively.

Dbisec is similar to *Tbisec*, but instead of using time, it uses the distance between feature vectors of consecutive frames as a measure of length in the bisection. The length of a sequence is taken to be the sum of the Euclidean distances between the consecutive frames composing that sequence. Therefore, a dynamic sequence containing more shots or more action will be "longer" than a static sequence with the same number of frames; it will be chopped into more pieces in the binary tree construction, and will contribute more frames in the summarization. An example of *Dbisec* summarization process is shown in Fig. 15.

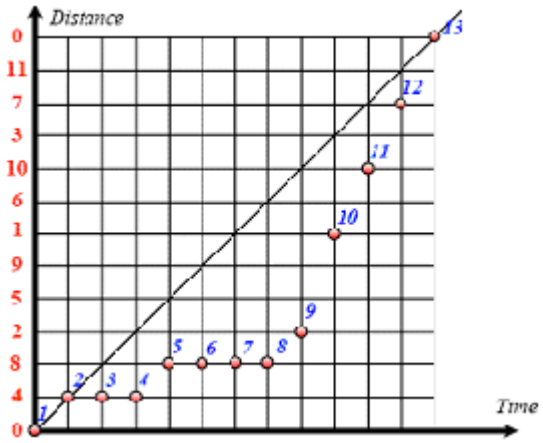


Figure 15: Distance bisection method for 13 frames. The frames are represented by circles. The number along each circle is the frame index. The beginning of the sequence is static, with slow distance change, and the end is dynamic, with large inter-frame distances. Bisection takes place along the vertical axis, and the order of choice of frames in the ranking process is given by the numbers along the vertical axis. Frames of the dynamic part of the sequence tend to be ranked higher than with a time-based bisection.

Our goal is to see whether ***Curve Simplification algorithm*** outperforms the other two in summarization ability and make corresponding improvements, so we tested several different types of feature vectors (See Section 3) and for the same feature vector construction scheme, we tried various numbers of bin or bin combinations to find out which feature vector performs best. To find out whether non-histogram based feature vector performs better, we also tried thumbnail image vectors C8, BW8, C16, and BW16, corresponding to RGB image after 8*8 block Gaussian smoothing, Gray image after 8*8 block Gaussian smoothing, RGB image after 16*16 block Gaussian smoothing, and Gray image after 16*16 block Gaussian smoothing. Thumbnail image vector is shortened as ***tImg*** in the tables.

7.3 Performance Comparison

As mentioned in Section 4.2.2, for frame ranking based algorithms, we need to set a cutoff point for the frame lists to compute the performance measures. There are several ways to this: one is to fix the sequence length; the other is to cut off the list when recall reaches a certain level. SUPERSIEV adopts the latter.

We have done experiments with cutoff point set at 100% recall and found it usually takes a long time for most candidates to locate the last few frames in the reference. But this is not a definite sign that all these candidates perform poorly, because an algorithm that quickly reaches a plateau at 99% of optimality may, in many applications, be preferable to an algorithm that has a guarantee of eventual optimality but a sluggish early performance improving rate. A more reasonable way is to compare performance curves (see Fig. 16) of each ranked list, but the results may be difficult to interpret. Further observation shows that the performance turning point (point after which the performance quickly gets worse) of most candidates is around 80% recall, so we

use this cutoff point to approximate curve-comparison. Table 1 to table 7 present the performance of each candidate summary. AC, CAP and N_{80} represent average closeness, cumulated average precision, and the number of frames it costs a candidate to reach 80% recall. To save space, results of redundancy rate are not presented; one can have a coarse estimate of redundancy by comparing N_{ref} with N_{80}.The entries in bold and italic style correspond to the best performance in the tables.

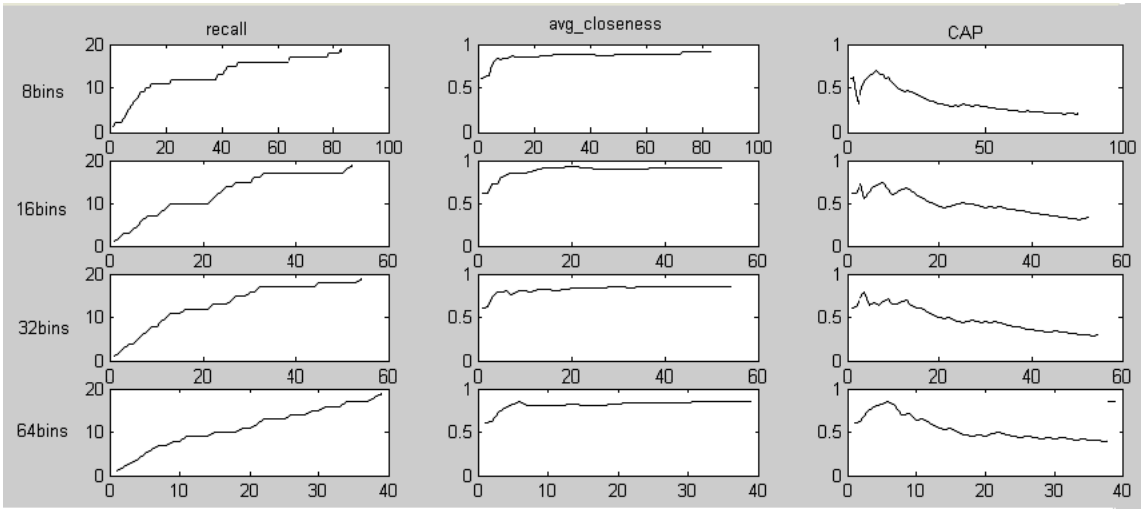


Figure 16: Performance of bY on WizzardDies.mpeg

Table 1: Performance of various bin combinations used in bY

nBin	**FlyingCar**			**Bebop**			**WizzardDies**		
	AC	**CAP**	$\mathbf{N_{80}}$	**AC**	**CAP**	$\mathbf{N_{80}}$	**AC**	**CAP**	$\mathbf{N_{80}}$
4	0.443	0.277	16	***0.819***	***0.512***	***32***	***0.876***	***0.532***	***28***
8	0.334	0.304	11	0.885	0.348	47	0.891	0.233	65
16	***0.578***	***0.481***	***12***	0.793	0.387	38	0.900	0.464	33
32	0.584	0.417	14	0.819	0.420	39	0.844	0.448	32
64	0.479	0.228	21	0.833	0.387	43	0.853	0.426	34
128	0.498	0.216	23	0.799	0.326	49	0.859	0.365	40
256	0.520	0.179	29	0.794	0.378	42	0.886	0.350	43

7.4 Observations and Discussions

1. **About videos**

   Generally, the average closeness (see column ***AC*** in the tables) of video *FlyingCar* is the lowest among the three videos, and that of *Bebop* is the highest, which is to be expected since the motion in *FlyingCar* is the most intense, while *Bebop*, a cartoon movie, has a large amount of static images.

Table 2: Performance of various bin combinations used in sY

nBin	**FlyingCar**			**Bebop**			**WizzardDies**		
	AC	**CAP**	$\mathbf{N_{80}}$	**AC**	**CAP**	$\mathbf{N_{80}}$	**AC**	**CAP**	$\mathbf{N_{80}}$
4	0.533	0.313	17	***0.854***	***0.534***	***32***	***0.861***	***0.488***	***30***
8	0.485	0.373	13	0.881	0.345	51	0.868	0.476	31
16	***0.578***	***0.481***	***12***	0.809	0.415	39	0.891	0.433	35
32	0.659	0.471	14	0.827	0.404	41	0.913	0.272	57
64	0.443	0.260	17	0.869	0.404	43	0.886	0.295	51
128	0.498	0.216	23	0.815	0.320	51	0.878	0.355	42
256	0.520	0.179	29	0.794	0.378	42	0.896	0.331	46

Table 3: Performance of various bin combinations used in 3DYUV

	FlyingCar			**Bebop**			**WizzardDies**		
	AC	**CAP**	$\mathbf{N_{80}}$	**AC**	**CAP**	$\mathbf{N_{80}}$	**AC**	**CAP**	$\mathbf{N_{80}}$
4,4,4	***0.529***	***0.352***	***15***	***0.807***	***0.425***	***38***	0.922	0.121	130
8,8,8	0.457	0.305	15	0.814	0.313	52	0.881	0.151	99
8,4,4	0.483	0.284	17	0.806	0.366	44	0.922	0.122	128
16,4,4	0.527	0.229	23	0.756	0.369	41	0.883	0.469	32
32,8,8	0.615	0.267	23	0.753	0.350	43	***0.890***	***0.473***	***32***

Table 4: Performance of various bin combinations used in bYUV

	FlyingCar			**Bebop**			**WizzardDies**		
	AC	**CAP**	$\mathbf{N_{80}}$	**AC**	**CAP**	$\mathbf{N_{80}}$	**AC**	CAP	$\mathbf{N_{80}}$
4,4,4	0.487	0.304	16	0.868	0.377	46	***0.877***	***0.532***	***28***
8,8,8	***0.347***	***0.315***	***11***	0.879	0.374	47	0.874	0.316	47
8,4,4	0.337	0.307	11	0.879	0.374	47	0.899	0.255	60
16,4,4	0.578	0.481	12	0.805	0.424	38	0.901	0.464	33
16,8,8	0.578	0.481	12	0.805	0.424	38	0.902	0.465	33
32,8,8	0.584	0.417	14	***0.822***	***0.444***	***37***	0.844	0.448	32
64,8,8	0.479	0.228	21	0.832	0.396	42	0.854	0.427	34

Table 5: Performance of various bin combinations used in sYUV

	FlyingCar			**Bebop**			**WizzardDies**		
	AC	**CAP**	$\mathbf{N_{80}}$	**AC**	**CAP**	$\mathbf{N_{80}}$	**AC**	CAP	$\mathbf{N_{80}}$
4,4,4	0.531	0.312	17	***0.860***	***0.538***	***32***	***0.861***	***0.488***	***30***
8,8,8	0.485	0.373	13	0.879	0.352	50	0.861	0.472	31
8,4,4	0.485	0.373	13	0.879	0.352	50	0.862	0.473	31
16,4,4	***0.578***	***0.481***	***12***	0.810	0.415	39	0.888	0.432	35
16,8,8	***0.578***	***0.481***	***12***	0.812	0.416	39	0.888	0.432	35
32,8,8	0.615	0.439	14	0.827	0.403	41	0.913	0.277	56
64,8,8	0.443	0.260	17	0.868	0.404	43	0.885	0.295	51

Table 6: Performance of thumbnail image vectors

	FlyingCar			**Bebop**			**WizzardDies**		
	AC	**CAP**	$\mathbf{N_{80}}$	**AC**	**CAP**	$\mathbf{N_{80}}$	**AC**	**CAP**	$\mathbf{N_{80}}$
C8	0.578	0.206	28	0.883	0.079	225	0.924	0.209	75
BW8	0.601	0.177	34	0.865	0.178	97	***0.824***	***0.369***	***38***
C16	***0.439***	***0.274***	***16***	***0.852***	***0.473***	***36***	0.914	0.239	65
BW16	0.507	0.203	25	0.893	0.372	48	0.889	0.226	67

Table 7: Best Performance Comparison

	FlyingCar			**Bebop**			**WizzardDies**		
	AC	**CAP**	$\mathbf{N_{80}}$	**AC**	**CAP**	$\mathbf{N_{80}}$	**AC**	CAP	$\mathbf{N_{80}}$
bY	***0.578***	***0.481***	***12***	0.819	0.512	32	***0.876***	***0.532***	***28***
sY	0.578	0.481	12	0.854	0.534	32	0.861	0.488	30
bYUV	0.347	0.315	11	0.822	0.444	37	***0.877***	***0.532***	***28***
sYUV	0.578	0.481	12	***0.860***	***0.538***	***32***	0.861	0.488	30
3DYUV	0.529	0.352	15	0.807	0.425	38	0.890	0.473	32
logRGB	0.501	0.186	27	0.811	0.386	42	0.873	0.464	32
tImg	0.439	0.274	16	0.852	0.473	36	0.824	0.369	38
TBisec	0.460	0.329	14	0.951	0.339	56	0.904	0.480	32
DBisec	0.540	0.385	14	0.788	0.450	35	0.872	0.449	33

2. **About assessors' behaviors**

   (a) for videos with many shots, assessors tend to choose one frame for each shot

   (b) very short shots are usually ignored

   (c) usually the first and last frames of a video, especially the last one, are ignored

   (d) for a shot with regular motion, such as the second shot of *FlyingCar.mpg*, key frames selected are quite evenly distributed in temporal dimension.

3. **About summarization algorithms**

   - Adding U, V channels affects little to the performance compared to using Y channel only; but combining UV with Y to form a 3D feature vector causes the performance to decrease slightly. This is because a bin count in the 3D color space is more sensitive to noise.
   - For basicY, when the number of bins is so small, different objects with similar luminance are put into one bin and performance is not good; with the increase of the number of bins within some range, performance goes up; then further increases will sometimes cause decrease of performance. This is because when the number of bins becomes much larger than the number of main objects, noise becomes an issue. Bin cleaning (assign 0 to bins whose number of pixels is below some threshold) might help remove the effect of noise.
   - The effect of Gaussian smoothing with diameter 33 on a 256-bin-histogram is not reliable, sometimes better, most of the times worse. This may be because histogram based spatial smoothing artificially changes the original image content, and usually the loss overshadows the gain.
   - logRGB is not the best, yet not the worst. More videos need to be added to the user study to give a clear view of the value of this kind of feature vector.
   - Feature vector from thumbnail frames perform rather poorly. And a 16*16-block-smoothed thumbnail generally performs better than an 8*8-block-smoothed one. The reason may be due to lack of as strong linear correlations between dimensions as other histogram-based feature vectors. In the latter there are strong linear correlations between bins, both number of pixels and centroid position of the bin population. This kind of linear correlation will amplify the image distance in some sense, e.g., change in one bin will cause changes in several other bins. And due to the application of Euclidean distance metric, which assumes dimension independence, this change is multiplied by a factor larger than 1. Because of the definition of relevance and the special scheme of curve simplification, this distance amplification works as an adaptive adjustment to image distance.
   - Both *Dbisec* and *Tbisec* perform well, with *Dbisec* outperforming the latter a little. It can be expected that with the increase of video length, *Dbisec*'s advantage will be more evident.

8 Conclusions and Future Work

We have developed a performance evaluation system, SUPERSIEV, for video summarization algorithms, which has three main characteristics: (1) automatic evaluation; (2) synthesizing reference summaries obtained from user study into one that expresses the majority of assessors' opinions; (3) using four performance measures to quantitatively evaluate a candidate summary from different view angles, which can help give a clear view about the behavior of the summarization algorithm and shed light on algorithm improvements. We also proposed to use an offline lookup table to accelerate the online performance computation.

To illustrate the operation of our system and how it is used to help analyze and compare algorithms, we evaluated three frame ranking based summarization algorithms. We have performed a large amount of experiments to help selecting appropriate feature vectors for curve simplification algorithm.

For the current evaluation system itself, there is still a lot of room for improvement, such as:

- Image content distance computed by means of feature vectors may sometimes contradict human's intuition, thus leading to wrong matching relationships. This is inevitable. Image distances that better model human cognition can be introduced in the framework of SUPERSIEV.

- Since the current evaluation scheme is shot based, it will treat two coherent shots, such as shots from the same scene, the same as incoherent shots. And because assessors are not required to choose at least one frame per shot, the final reference summary is not a shot based one. One way to deal with this inconsistency is to make sure that the final reference has at least one frame for each shot. High level video segmentation techniques, e.g., scene clustering, can help lessen this shot boundary dependency, but are more difficult to implement.

- It is worth finding a way to align different videos (shot number, video length, genre, etc.), so that we could give a more global mark on the quality of an algorithm by averaging its performance over different videos.

- For frame ranking algorithms, the current system still falls short of proposing an effective means to evaluate the correctness of individual rankings of frames, because assessors are not required to rank frames (this would be too much work), the same problem as in TREC-type evaluation systems.

In addition, our current system is based on low level features, so it inherits all the drawbacks of video applications using low level features. Finally, algorithm complexity and robustness are not evaluated, since our present focus is only on the summarization process.

References

[1] D. DeMenthon, V. Kobla and D. Doermann, "Video Summarization by Curve Simplification", ACM Multimedia 98, Bristol, England, pp. 211-218, September 1998.

[2] D. DeMenthon, L. Latecki, A.Rosenfeld and M. Vuilleumier Stckelberg, "Relevance Ranking of Video Data using Hidden Markov Model Distances and Polygon Simplification", VISUAL 2000, pp. 49–61.

[3] A. Divakaran, K.A. Peker, R. Radhakrishnan, Z. Xiong and R. Cabasson, "Video Summarization using MPEG-7 Motion Activity and Audio Descriptors", Video Mining, A. Rosenfeld, D. Doermann et D. DeMenthon, eds., Kluwer Academic Press, 2003.

[4] M.S. Drew and J. Au, "Video Keyframe Production by Efficient Clustering of Compressed Chromaticity Signatures", ACM Multimedia 2000, Juan-Les-Pins, France, pp. 365–368, November 2000.

[5] Y. Gong and X. Liu, "Video Summarization using Singular Value Decomposition", CVPR 2000, pp. 2174–2180.

[6] A. Hanjalic and H. Zhang, "An Integrated Scheme for Automated Video Abstraction based on Unsupervised Cluster-Validity Analysis", IEEE Transactions on Circuits and Systems for Video Technology, Vol.9, No.8, pp.1280–1289, 1999

[7] L. He, E. Sanocki, A. Gupta, and J. Grudin, "Auto-Summarization of Audio-Video Presentations", ACM Multimedia 99, Orlando, vol. 1, pp. 489–498, 1999.

[8] Y. Li, T. Zhang and D. Tretter, "An Overview of Video Abstraction Techniques", HP Laboratory Technical Report, HPL-2001-191, July 2001.

[9] S.B. Needleman and C.D. Wunsch, "A general method applicable to the search for similarities in the amino acid sequence of two proteins", J. Molecular Biology, vol. 48, no. pp. 443-453, 1970.

[10] K. Papineni, S.Roukos, T. Ward, W-J. Zhu, "Bleu: a Method for Automatic Evaluation of Machine Translation", ACL 2002, pp. 311–318.

[11] K. Pastra and H. Saggion, "Colouring Summaries BLEU", EACL 2003.

[12] M. Slaney, D. Ponceleon and J. Kaufman, "Understanding the Semantics of Media", Video Mining, A. Rosenfeld, D. Doermann et D. DeMenthon, eds., Kluwer Academic Press, 2003.

[13] H. Sundaram, L. Xie, S-F. Chang,"A utility framework for the automatic generation of audio-visual skims", ACM Multimedia 2002, pp.189–198.

[14] S. Uchihashi, J. Foote, A. Girgensohn and J. Boreczky, "Video Manga: Generating Semantically Meaningful Video Summaries", ACM Multimedia 99, Orlando, vol. 1, pp. 383–392, 1999.

[15] J. Vermaak, P. Perez, M. Gangnet and A. Blake, "Rapid Summarization and Browsing of Video Sequences", BMVC 2002, pp. 424–433.

[16] E. M. Voorhees, "Variations in relevance judgments and the measurement of retrieval effectiveness", Information Processing and Management, vol. 36, no. 5, pp. 697–716, 2000.

[17] H. Wactlar, M. Christel, Y. Gong and A. Hauptman,"Lessons Learned from Building a Terabyte Digital Library", IEEE Computer, Vol. 32, No. 2, pp. 66–73, 1999.

[18] K. Yoon, D. DeMenthon and D. Doermann, "Event detection from MPEG video in the compressed domain, ICPR 2000, Vol. 1 , pp. 819–822.

[19] Yu-Fei Ma, Lie Lu, Hong-jiang Zhang and Mingjing Li, "A User Attention Model for Video Summarization", ACM Multimedia 2002, Juan-les-Pins, France,December, 2002

Appendix

A Assumptions about Assessors Behavior

The reference synthesis procedure is based on two assumptions about assessors' behaviors:

1) Each assessor has his/her own stable semantic gap in consecutive summary frames; which can be viewed as a constant in one video. — That is, the reference summary from one assessor will not give viewers a feeling that the change between the content of some neighboring frames jumps sharply while in other parts the content goes on continuously and smoothly. Under this assumption, we can define the semantic gap that most assessors agree upon as the main stream gap and normalize each assessor's summary by inserting frames for a sparse style or combining several frames into one for a dense style. We then can cluster all normalized reference summary frames into several groups with the gap between centers of neighboring groups along the time dimension close to the main stream semantic gap. The resultant summary, composed of group centers, can be viewed as consistent with the majority opinion of assessors.

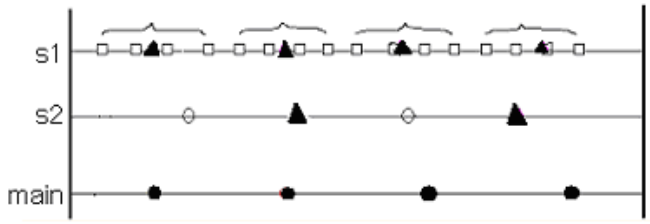


Figure 17: Assumption of constant semantic gap for an assessor.

For example, in Fig. 17, suppose the lines represent the semantics axis (equal intervals means equal semantic gaps). The squares and circles on the first two lines correspond to reference frames from two assessors (S_1 and S_2) respectively. And the filled circles on the third line correspond to the main stream summary of all assessors, which is 4 frames in length. Given a main stream summary, we can normalize those summaries (S_1 and S_2 in this case) whose style (semantic gap) is ***so different*** from average, based on the above assumption: combining every 4 frames into one forS_1, and insert two frames for S_2 (results are marked as filled triangles).

The problem is that what we discuss here is based on an ideal semantics axis, whose implementation involves semantic level content representation and distance computation, but at present there is no mature technology to deal with this. So, we alternatively try to figure out the summary

length (number of frames) that most assessors agree upon as the expected number of clusters and use an iterative procedure to model the main stream summary in low level image feature space.

2) Assessors have some common judgment about the current video. That is, the following situation is defined as weird and is not assumed to occur in real cases. In Fig. 18, S_1 thinks the former part of the video is very important and reflects the whole content, while the rest of the video clip is insignificant. And S_2 and S_3 have very different feelings. If we allow this to happen, the main stream summary obtained by using the above method is in fact an incorrect model of assessors' summaries. It is a simple add-up of the three summaries and none of the assessors would agree it is representative of his/her opinion.

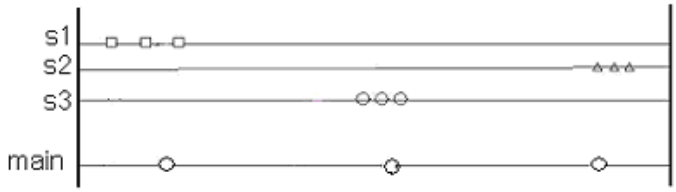


Figure 18: When no common opinion about the video content can be derived, no main stream summary can be extracted. The so called main stream summary in this case in fact does not consist with any assessors' summaries.

We assume that video producers (movie makers) will not waste films on producing **a large amount** of trivial frames, i.e. each frame does contribute to the whole content. So the above situation means the assessors have evidently forced their personal interests on their summaries and the completeness of the video content is not preserved, which are against our guidelines.

Note there does exist situations such as Fig. 19 shows, in which most assessors select one key frame. This implies that the information amount of the segment is low, and assessors think that one frame is enough for abstraction. This is typical for a stationary video segment. For whom that chooses two frames (S_2), we think he/she does not follow our suggestion; this is a minority case which will not impede the main stream summary extraction.

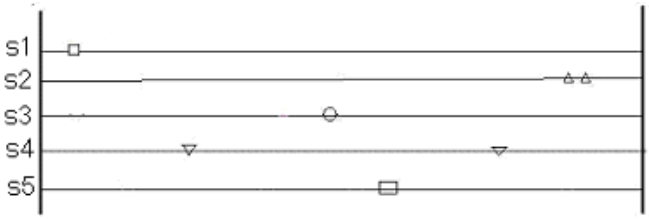


Figure 19: Typical behavior of assessors for a stationary video segment

Though both figures show large difference between assessors, the main difference between Fig. 18 and Fig. 19 is that in the former case, assessors choose more than one frame at very close positions, that is, they make dense sampling locally, which cannot be regarded as a random selection. And these local dense clusters are far away from each other, implying large cognitive divergence between assessors. While in the latter case, the randomly distributed key frames are a

good sign that the information distributes relatively even in the segment and assessors agree that the segment can be summarized by one frame. This, of course, is a kind of cognitive convergence. When no mainstream opinion of assessors can be extracted, evaluation cannot be conducted. So this second assumption of consistent common judgment is to provide a condition such that reasonable mainstream reference does exist.

B Performances at 100% Recall

Table 8 to Table 14 show the performance at 100% recall. The column $\boldsymbol{N}_{100}$ is the number of frames it costs the candidate summary to reach 100% recall.

Table 8: Performance of various bins used in bY

nBin	**FlyingCar**			**Bebop**			**WizzardDies**		
	AC	**CAP**	$\mathbf{N}_{100}$	**AC**	**CAP**	$\mathbf{N}_{100}$	**AC**	**CAP**	$\mathbf{N}_{100}$
4	0.594	0.174	41	0.977	0.095	247	0.903	0.330	52
8	***0.424***	***0.318***	***16***	0.960	0.135	170	0.910	0.208	83
16	0.593	0.309	23	***0.888***	***0.152***	***140***	0.910	0.333	52
32	0.528	0.334	19	0.876	0.146	144	0.857	0.301	56
64	0.598	0.217	33	0.961	0.065	357	***0.857***	***0.417***	***39***
128	0.592	0.203	35	0.900	0.092	234	0.866	0.350	47
256	0.681	0.132	62	0.946	0.060	378	0.890	0.319	53

Table 9: Performance of various bins used in sY

nBin	**FlyingCar**			**Bebop**			**WizzardDies**		
	AC	**CAP**	$\mathbf{N}_{100}$	**AC**	**CAP**	$\mathbf{N}_{100}$	**AC**	**CAP**	$\mathbf{N}_{100}$
4	***0.532***	***0.278***	***23***	0.915	0.154	143	0.913	0.231	75
8	0.595	0.216	33	0.942	0.138	164	***0.878***	***0.464***	***36***
16	0.627	0.313	24	0.932	0.100	223	0.898	0.371	46
32	0.600	0.240	30	***0.880***	***0.215***	***98***	0.920	0.208	84
64	0.605	0.214	34	0.955	0.101	228	0.919	0.196	88
128	0.539	0.202	32	0.908	0.096	227	0.888	0.318	53
256	0.681	0.134	61	0.946	0.064	357	0.906	0.233	74

Table 10: Performance of various bin combinations used in 3DYUV

YUV	**FlyingCar**			**Bebop**			**WizzardDies**		
	AC	**CAP**	$\mathbf{N_{100}}$	**AC**	**CAP**	$\mathbf{N_{100}}$	**AC**	**CAP**	$\mathbf{N_{100}}$
4,4,4	0.754	0.127	71	0.937	0.097	231	0.937	0.102	175
8,8,8	0.689	0.145	57	0.929	0.125	178	0.947	0.094	191
8,4,4	0.744	0.131	68	0.935	0.117	191	0.930	0.122	145
16,4,4	0.659	0.149	53	0.931	0.087	256	0.908	0.227	76
32,8,8	***0.628***	***0.209***	***36***	***0.926***	***0.139***	***160***	***0.889***	***0.393***	***43***

Table 11: Performance of various bin combinations used in bYUV

YUV	**FlyingCar**			**Bebop**			**WizzardDies**		
	AC	**CAP**	$\mathbf{N_{100}}$	**AC**	**CAP**	$\mathbf{N_{100}}$	**AC**	**CAP**	$\mathbf{N_{100}}$
4,4,4	0.619	0.186	40	0.963	0.102	226	0.902	0.330	52
8,8,8	***0.435***	***0.326***	***16***	0.960	0.134	172	0.915	0.195	89
8,4,4	0.427	0.320	16	0.960	0.135	171	0.917	0.198	88
16,8,8	0.554	0.289	23	0.888	0.152	140	0.911	0.333	52
16,4,4	0.554	0.289	23	***0.886***	***0.153***	***139***	0.911	0.333	52
32,8,8	0.548	0.346	19	0.900	0.150	144	0.857	0.301	54
64,8,8	0.598	0.199	36	0.961	0.064	359	***0.859***	***0.419***	***39***

Table 12: Performance of various bin combinations used in SmoothYUV/sYUV

	FlyingCar			**Bebop**			**WizzardDies**		
	AC	**CAP**	$\mathbf{N_{100}}$	**AC**	**CAP**	$\mathbf{N_{100}}$	**AC**	**CAP**	$\mathbf{N_{100}}$
4,4,4	0.529	0.276	23	0.914	0.156	141	0.913	0.259	67
8,8,8	0.595	0.216	33	0.940	0.137	165	0.872	0.460	36
8,4,4	0.595	0.216	33	0.940	0.138	163	***0.873***	***0.461***	***36***
16,8,8	0.627	0.313	24	0.920	0.099	222	0.895	0.370	46
16,4,4	0.627	0.313	24	0.932	0.100	223	0.895	0.370	46
32,8,8	***0.554***	***0.350***	***19***	***0.879***	***0.209***	***101***	0.920	0.206	85
64,8,8	0.605	0.214	34	0.955	0.101	228	0.919	0.194	90

Table 13: Performance of thumbnail image vector

	FlyingCar			**Bebop**			**WizzardDies**		
	AC	**CAP**	$\mathbf{N_{100}}$	**AC**	**CAP**	$\mathbf{N_{100}}$	**AC**	**CAP**	$\mathbf{N_{100}}$
C8	0.684	0.178	46	0.902	0.039	556	0.925	0.212	83
BW8	0.586	0.176	40	0.961	0.053	438	***0.858***	***0.226***	***72***
C16	***0.546***	***0.211***	***31***	0.893	0.128	167	0.912	0.217	80
BW16	0.532	0.177	36	***0.943***	***0.166***	***136***	0.907	0.185	93

Table 14: Best Performance Comparisons

	FlyingCar			**Bebop**			**WizzardDies**		
	AC	**CAP**	$\mathbf{N_{100}}$	**AC**	**CAP**	$\mathbf{N_{100}}$	**AC**	**CAP**	$\mathbf{N_{100}}$
bY	0.424	0.318	16	0.888	0.152	140	0.857	0.417	39
sY	0.532	0.278	23	***0.880***	***0.215***	***98***	***0.878***	***0.464***	***36***
bYUV	***0.435***	***0.326***	***16***	0.883	0.151	139	0.859	0.419	39
sYUV	0.554	0.350	19	0.879	0.209	101	0.873	0.461	36
3DYUV	0.628	0.209	36	0.926	0.139	160	0.889	0.393	43
logRGB	0.622	0.097	77	0.909	0.130	168	0.882	0.399	42
tImg	0.546	0.211	31	0.943	0.166	136	0.858	0.226	72
TBisec	0.532	0.266	24	0.907	0.087	249	0.914	0.386	45
DBisec	0.658	0.255	31	0.897	0.353	61	0.952	0.152	119